## **Amendments to the Specification:**

Please replace the paragraph beginning on page 6, line 28 and ending on page 7, line 2 with the following amended paragraph:

FIG. 3 illustrates conceptual interrelationships between the text object model in accordance with an embodiment of the invention and an application 145. Basically the operations 300 in this embodiment of the invention may be viewed as an abstraction between user interactions 302 and a text store [[304]] 306. The user interactions 302 may each include such operations as typing, rendering, defining layout of the object and editing the content of the object. The abstraction generally, may be viewed as a text object model [[306]] 304 that receives direction and interfaces with the user interactions 302 and in turn draws from one or more text stores [[304]] 306 to accomplish the actions required.

Please replace the paragraph beginning on page 7, line 11 and ending on page 7, line 21 with the following amended paragraph:

The application program interface 155, i.e. text object model 400 in accordance with an embodiment of the present invention has two fundamental parts - data model 156 and view model 158. Data model 156 provides access to persistent content of text. For example, these include characters, embedded objects, and formatted and structuring elements. The view model 158 provides access to presentation and interaction appearance of text: lines and other layout blocks, dynamic highlights of various kinds, such as selection, misspellings, carets, etc. This model is represented by [[an]] a non-abstract class "TextView" and an abstract class "TextHighlight". Note that TextHighlight is actually part of the data model. The view model is a superset of the data, adding additional layout functionality such as hit testing. TextHighlight is data that serves as input to the layout engine, just like TextContainer. These models are provided in the backing store of the API 155.